From: AAAI Technical Report FS-00-03. Compilation copyright © 2000, AAAI (www.aaai.org). All rights reserved.

AI approaches to developing strategies for 'wargame' type simulations

Vincent Corruble

Laboratoire d'Informatique de Paris 6 (LIP6)
Université Pierre et Marie Curie (Paris 6)
4, place Jussieu
75252 Paris Cedex 05
France
Vincent.Corruble@lip6.fr

Abstract

This article describes an ongoing project that aims at developing a strong AI-based artificial opponent for some wargame-type simulations. It describes the two approaches taken so far. The first one relies mainly on an incremental and automated learning of the complex knowledge necessary to devise sound tactics, but needs a high-quality coaching which itself requires in-depth knowledge of the domain. The second approach is an attempt at modeling the decision making process that goes on at the various levels of a entire army as some form of complex and knowledge-intensive problem-solving. Future work will look at the possibility of hybrid methods as well as at some alternative methods such as case-based reasoning.

Motivation

The history of AI involvement in games is long and rich; it is not the place to show in detail its significance for the discipline. From the Turing test, an imitation *game*, to early work on chess and checkers (Newell and Simon 1963, Samuel 1959), games have provided many of the world models in which AI ideas and programs were developed and tested. In the 70s and 80s, a shift occurred toward real-world applications; it appeared necessary to show that AI was a mature science. It was then believed that only real-world applications (e.g. in medicine, etc.) would provide the required complexity, while games would remain merely toy problems for the discipline.

This is not as simple as that anymore. Computer games have grown more complex, and even though there is still ground to cover before they display the type of complexity we humans tackle in living in the real world, many computer games show a level of complexity which compares with, sometimes exceeds, the one of many current 'real-world' applications of AI.

We have embarked on a project which aims at trying a number of AI approaches on a type of computer games simulating with a certain level of historical realism military tactics of the previous centuries. The main motivation is to test and improve these AI techniques. Also, as a practical result, some of these games could be provided with a worthy artificial opponent. Most of these games already provide the option of playing 'against the computer': it means typically that the human player plays one side while the software plays the other. Curiously this virtual/artificial opponent is named 'the AI' by the gaming community, in large part because software publishers often boasts that their system uses some AI-based technology. However our limited experience shows this 'AI' often shows limited performances to say the least. For one thing, it shows a very limited ability to adapt and to improve.

In the remaining of this paper, we first introduce the main characteristics of the type of games that we are studying. Then we will introduce the range of AI techniques which are being tried, organized by increasing requirements in terms of background knowledge (i.e. starting from pure learning techniques up to knowledge based systems). Then we will conclude with plans for experimentation, which have not been conducted yet.

The strategy games

Our work has so far focused on a well-established commercial game that simulates some key battles of Napoleon's times. There exists a strong community of players for this game, and therefore some serious outside expertise will be available to test our system when it is ready. Additionally, another simpler game designed by students of UPMC, similar in many respects, but focusing on medieval tactics instead of Napoleon, has been used in a first stage to test some ideas, in particular on the use of SOAR.

Rather than going into the details of the rules of these two games, we can illustrate the basic game concepts by emphasizing their differences with the classic game of chess (chess can also be seen as modeling –or rather symbolizing- military tactics). As in chess, each player plays a side in the conflict. The theater is represented by a board, and each player plays in turn by moving units on these boards according to a number of rules. The differences can each be seen as an added complexity compared to the game of chess.

The board in our games is not made of squares but of hexagons. This allows for more complex movement patterns (each location as 6 neighbors). More importantly, while the board contains only 64 squares in the game of chess, it counts thousands of hexagons in the games we consider here.

Each hexagon is defined, on top of its location, by a number of parameters, such as the type of terrain (forest, plain, ...), the elevation, etc. which in turn affect the units ability to move or fight.

Each unit is defined by a number of characteristics, which affects its ability to move, to fight, as well as its quality, fatigue, organization, etc. Some of them (such as the unit configuration in line or column) can be seen as unit parameters that affect the unit's abilities, such as moving or fighting.

Units are organized in a hierarchy (such as Army, Corps, Division, Brigade...). The force cohesion requires that units belonging to the same organization fight together.

Each game turn is divided in a number of phases which are each devolved to a type of activity, such as movement, range fire, etc.

Combat itself is modeled in a relatively complex fashion. It includes range fire (fire from a distance by infantry with muskets or artillery) and melees (hand to hand combat). Its resolution takes into account many parameters including the size, quality, status, fatigue, etc. of each side. It also includes a stochastic component so that the outcome is (almost) never guaranteed.

A game is won by the side that scores the highest number of points in a given amount of time (i.e. in so many game turns). Points are obtained by seizing (or successfully defending) key locations, and by putting out of action enemy units.

Maybe a more significant difference between our games and the game of chess, from the computational point of view, is that at each game turn every one of the units can be active. This departs radically from chess where each turn sees each side choose a unique piece to move. The resulting increase in complexity is very significant: If there are p units in the games, each with m possible actions, the resulting branching factor is $p.m$ for chess, while it is m^p for our games[1]. Besides an explosive complexity, this different setup has the advantage of allowing a simulation that is much closer to history, or for that matter to the simulation of human agents, the theme of this symposium. Army units do move in a parallel fashion around the theater so if historical plausibility is a concern (and it is in our case), sequential action (as in chess) is out of question.

As a result, while blind search methods can be envisaged to play chess to a certain level, this approach is just inconceivable for our games, since even generating all the possible states one turn forward is out of the question (for m=20, p=100, $1.28.10^{130}$ states would have to be generated...). So even more than for chess, and as for most 'real' AI problems, everything rests in the quality of the knowledge provided to –or learned by– the system.

1 To make things 'worse', our games typically involve hundreds of units on each side while it is limited to 16 in chess.

Active learning approaches

The first approach we have been experimenting with is one that has the lowest requirements in terms of background knowledge. The idea is to start by simulating small-scale confrontations and simple objectives so that individual units can learn some basic knowledge, and then to learn incrementally more complex knowledge by simulating more complex situations. The first learning method we are experimenting with is the genetic-based learning of rule bases, and more specifically a method inspired from Wilson's work on ANIMAT (Wilson, 1987, Meyer et al., 1997).

Different setups are tried, some using only basic sensory information (besides the unit goals, information about immediate surroundings, including the presence of enemy units), and some representing in the genetic code some intermediary concepts such as the perceived local force ratio (directly implying the notion of threat). After these initial stages, additional rule bases can be obtained by having the units work in larger and larger formations, in 'coaching scenarios' of increasing complexity.

This approach is very exciting since it is the most likely to provide surprising results (because it is less biased by our *a priori* on what would be an appropriate strategy for a given situation). Surprise could be the result of some form of strategic creativity shown by our systems. It could also mean that the rules of the game are unrealistic and that our method was better than others (human or artificial) at finding the loopholes in these rules[2].

This incremental approach to the learning of classifier systems should improve our chance that the knowledge bases eventually converge. However it is too early at this stage to conclude that they will actually converge. A significant difficulty here lies in the design of appropriate training scenarios of increasing complexity that can let the system learn basic concepts and tactics and then more advanced strategies. It has appeared after our initial experiments that this stage itself requires some in-depth expertise on the game. Hence, though the learning method seemed at first less demanding in terms of background knowledge (and it is in many respects), it appears that it would benefit greatly from all the lessons learned in the phases of modeling and knowledge development of the 'knowledge-intensive' approach.

Knowledge intensive approach

At the other end, we are using a knowledge intensive approach by eliciting the relevant knowledge within the

2 This would indeed correspond to a significant and well-known occurrence in AI history, i.e. when Lenat's Eurisko (Lenat, 1983) system found a completely unrealistic but nonetheless extremely efficient strategy for Traveler TCS, a simple simulation of naval battles. In turn, finding loopholes in the rules of the game can be of a *great* interest... to the game designers!

SOAR (Laird et al., 1991) framework. SOAR has proved over the years to be a sound framework to represent and use complex knowledge within a dynamic environment. It has also more recently been successfully applied to a number of computer games, after some application to military simulation and planning. The difficulty here resides in the formulation of good rules for the knowledge bases, and to the choice of relevant intermediary concepts. Also it is particularly important to make good use of the ability to learn by chunking within SOAR since our initial intuition was that current systems are particularly limited by their inability to learn.

It would be possible to see each unit as a different agent. Indeed the game lets the player decide (within the constraints of the current situation) the action of each unit (infantry battalion, cavalry squadron for instance), as would the corresponding leader. However, the problem solving activity of each unit leader makes sense only in the context of the global strategy devised by higher-ranking leaders (at the divisional, corps, and army levels). Hence the information flow is bi-directional: on one hand it goes upward (low-level leaders report on what they observe, and on the success/failures of their actions) while on the other hand information goes downward: strategies are chosen, orders given, interpreted and as far as possible implemented, sometimes after some adaptation.

It was observed that this hierarchical information flow bears an interesting resemblance with the way goals and operators are organized in SOAR. Hence the design choice has been so far to consider an entire army as a single complex problem-solving agent whose hierarchical structure is largely reflected in the SOAR operator hierarchy. As a consequence of this choice, it has been decided to share knowledge bases between units of the same type, therefore not allowing 'individual unit learning' but only learning by unit type. Of course each level in the hierarchy of the unit organization does its learning independently from lower or higher levels.

Also the game has a special coaching mode where the user has the ability to suggest a course of actions to the program when its current knowledge base is either too poor or inadequate.

Future work

Another approach that we wish to investigate is the use of some forms of lazy learning for our games. This approach would place itself in between the two previous ones in terms of requirements in a priori knowledge. Case-based reasoning does need some knowledge in the choices relative to the representation of the cases, the organization of the case base, the similarity measure used for case retrieval, and the strategies for adaptation. Therefore the requirements are higher than for the genetic-based learning approach, but lower than for the knowledge intensive approach. This approach will have the advantage of proposing as solutions to potential situations entire plans of actions that will have succeeded in the past. Again, different case bases will be required for the different levels of the organization. An important element to take into account in the calculation of the similarity between cases is the similarity between the units' objectives since similarity between situations (geographical and surroundings) is not enough to motivate the retrieval of a case.

Besides trying more techniques, future work will require combining the most promising ones in hybrid systems that can rely on the technique the most adapted to the current situation. Since all the techniques seen here rely on some element of learning, it is necessary to allow some form of experimentation where the system can play against itself at high speed to reach some basic performance level. For the active learning approach, the method will be greatly enhanced by having the system play against the knowledge intensive method (even if its knowledge base is simplistic at first). The idea here is that the active learning method will start as a 'random' player (before anything is learned) and would therefore not show much progress by playing against itself. Additionally, the knowledge acquired with the knowledge-intensive approach will be helpful to design better coaching scenarios for the incremental active learning approach.

Experimental results on the two first approaches introduced in this paper are expected by the time of the symposium.

Acknowledgements

A number of CS students of the University Pierre et Marie Curie played active parts in the initial stages of this project. I wish to thank Louis Montagne and Pierre Falticska (from the 'DESS IA' graduate program) as well as the following undergraduate students: Gilles Bénichou, José De Sousa, Fabien Flacher, and Fabrice Serrière.

References

Laird, J.E., Newell, A., Rosenbloom, P.S. 1991. Soar: An Architecture for General Intelligence. Artificial Intelligence, 47: 289-325

Lenat, D. B. Eurisko: A program which learns new heuristics and domain concepts. Artificial Intelligence, 21, 1983.

Meyer, C., J.-G. Ganascia, and J.-D. Zucker. (1997). Learning Strategies in Games by Anticipation. International Joint Conference on Artificial Intelligence (IJCAI), Nagoya, Japan

Newell, A., & Simon, H.A. (1965). An example of human chess play in the light of chess playing programs. In N. Wiener and J.P. Schade (Eds.), Progress in biocybernetics (Vol. 2, pp. 19-75). Amsterdam: Elsevier.

Samuel, A. Some studies in machine learning using the game of checkers. IBM Journal of Research and Development, 3(3):211--229, 1959

Wilson, S.W. 1987. Classifier Systems and The Animat Problem. Machine Learning, 2 (3), 199-228.